School of Electronic
Engineering and
Computer Science

MSc Big Data Science
Project Report 2018

# RGB-D image semantic segmentation using deep learning

Ekaterina Romanovna
Lyapina


August 2018

**Abstract**:

In this paper the task of semantic image segmentation with Deep Learning is addressed, presenting a transfer learning approach for image segmentation of SUN RGB-D dataset. This work shows how to re-train Mask R-CNN on a new dataset and provides convenient framework to work with SUN RGB-D dataset and Kinect v2 integration. This problem is extremely important for robotics manipulation, as it represents the first step to successfully grasp and manipulate objects in unstructured environments, provide the necessary task-relevant information and allow to do tasks more reliably and efficiently. The results shows which performance the Mask-RCNN algorithm can achieve just fine-tuned on another RGB-D dataset, without implementing the depth channel. Experimental evaluations demonstrate that the proposed architecture achieves 10% mAP on NYU dataset and 18% mAP on the challenging SUN RGB-D benchmark dataset with the minimum fine-tuning of the network's structure. All the code is available online.

# 1 INTRODUCTION

Image segmentation aims to five a class label for each pixel on the image according to its semantic meaning. This problem is one of the most challenging tasks in image processing and is has received a lot of attention from AI community. Until recently methods using hand-crafted features such as HOG [1], SIFT [2], and SURF [3] were mostly used for image processing. A lot of prior knowledge is required in order to create efficient features, and the accuracy mostly depends on the researcher's skills. The current state-of-the-art approaches mostly use Convolutional Neural Networks (CNNs), instead of hand-crafted features in the image segmentation task, which achieve limited accuracy.

Depth information has a potential to improve the image segmentation results via adding more complementary data about the image to a backbone model. The depth channel can be captured with low cost RGB-D sensors like Kinect. Depth sensors have become more precise and affordable. There is a need to admit, that the task of object detection can be easily done without the depth channel, just based on the colour and texture attributes. Depth channel has a potential to distinguish objects having similar appearance information. This is crucial in many perceptual applications including robotics. In general, indoor scenes have rich semantic information, but they are more challenging than outdoor scenes due to more severe occlusions of objects and cluttered background. There is where depth comes into play.

Many state-of-the-art algorithms are mostly focused on RGB datasets. Consequently, the first step in solving the problem of RGB-D image segmentation is implementation of modern approaches for RGB-D datasets. In this work the potential performance applying the state-of-the art algorithm Mask RCNN [4] to the existing well-known SUN RGB-D [5] dataset solving the task of image segmentation was investigated. Pretrained on the COCO dataset [6] Mask RCNN model was fine-tuned using full SUN RGB-D dataset as well as using it's part - NYUv2 dataset [7].

One of the purposes of research project is to provide the framework for handling the SUN RGB-D dataset and converting existing annotations to more popular VGG Image Annotation format. The mapping scheme and conversion script was created to extract all labels from SUN RGB-D toolbox and make them VGG annotator's style, which potentially might stimulate a further use of this dataset. In this work the mapping scheme for 13 classes was used, despite mapping to 37 and 40 classes were also developed. The input for Mask RCNN model is the annotated datasets, converted to the friendly VGG format.

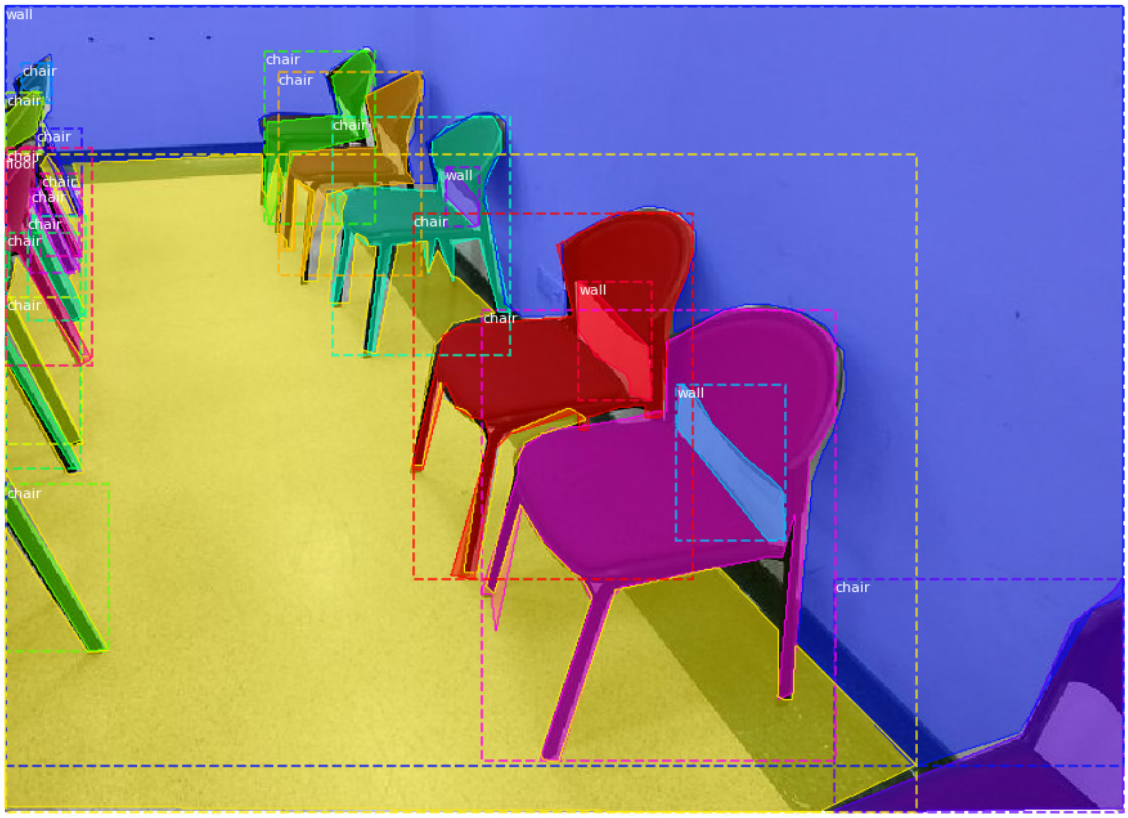


Figure 1: Example of an annotated image from SUN RGB-D dataset

The output of the model consists of bounding boxes, instance masks and class assignment. For the sake of the demonstration, integration between TensorFlow models and Kinect v2 was developed.

# 2 RELATED WORKS

## 2.1 The State of the Arts on Semantic segmentation

Traditionally, sliding window method was used to identify objects on an image. The problem was, that this method produces too many windows to analyse. Recent researches show, that objects have common certain properties differentiating them from the background, which might be used for object detection and localization. This object proposals are more effective in terms of computational efficiency and the quality of object detection.

R-CNN [8] pioneered this field of research, followed by Fast R-CNN [9] and Faster R-CNN [10]. The region-based approaches aim to assign a reasonable number of candidate object regions and evaluate convolutional networks independently on each region of interest (RoI). Because CNN does not need to analyse the whole image, this lead to Improvement in both speed and accuracy. For the image segmentation task, 2 types of features are extracted for each region by R-CNN: full region features and foreground features.


Figure 2: Example of an RoI on image from SUN RGB-D dataset

The most widely used algorithm for doing semantic segmentation is a FCN model [11], which is a pixel-to-pixel trained convolutional network, which re-purposes ImageNet pretrained networks for segmentation (AlexNet [12], the VGGnet [13], and GoogLeNet [14]).
FCN transfers knowledge from the abovementioned networks to perform semantic segmentation using 1x1 convolution. This process produces a class presence heat map in low resolution and up-sample them using deconvolution layers and unpooling.

Mask RCNN [4] combines two networks: Faster R-CNN (with replaced initial VGG-16 into ResNet-101 [15] backbone) and FCN in one meta model. The loss function for the model is the combination of several losses: for generating masks, bounding boxes and doing classification.

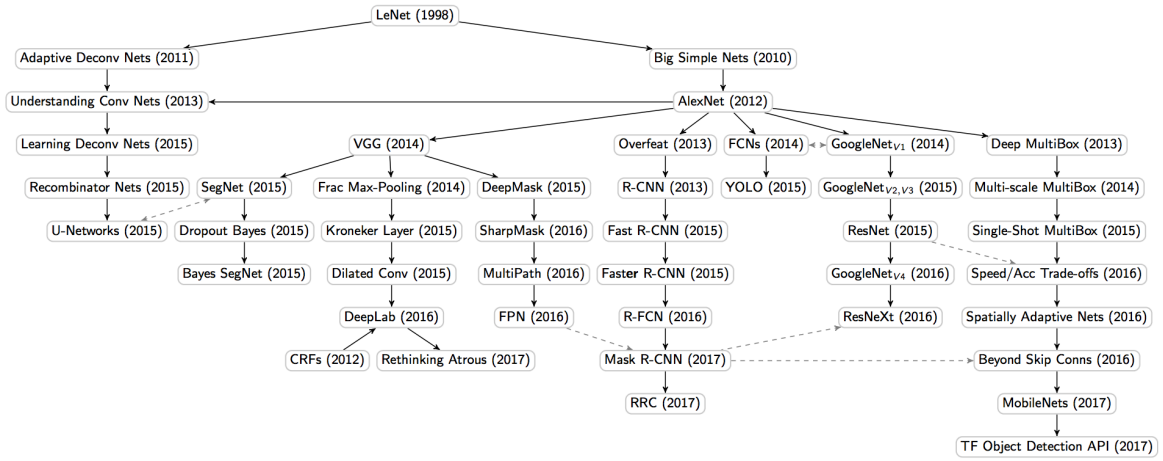


Figure 3: Family tree of different models

### 2.2 The State of the Arts on RGB-D Data

There are some studies focused on semantic segmentation task. It was observed [16] that the segmentation of classes having similar depth, appearance and location is improved by making use of the depth information. However, it is better to use only RGB information to recognize object classes containing high variability of their depth values. object volumes and scene layout extracted from the depth channel might successfully complement RGB channel in some cases

Previous works in RGB-D image segmentation combined two networks for RGB and depth data. The common approach is to pretrain model with a large RGB dataset and then fine tune it on the target RGB and depth dataset. These approaches have several limitations: 1) only use low-level filters learned from RGB data, thus not being able to exploit properly depth-specific patterns, and 2) RGB and depth features are only combined at highlevels but rarely at lower-levels.

One of the most successful implementation might be considered a FuseNet [17] network, which seems to tackle these issues of semantic labelling of indoor scenes on SUN RGB-D dataset. Authors found out a solution for incorporating the depth information into an encoder-decoder type fully convolutional CNN. The encoder part is composed of two branches of networks that simultaneously extract features from RGB and depth images and fuse depth features into the RGB feature maps as the network goes deeper.

Gupta et al. [18] presented encoding of depth information into RGB, which became a popular approach, names referred as HHA. It comprised horizontal disparity, height from the ground and the angle between the local surface normal and gravity direction. Experimentation proved use of HHA encoding helps. This approach was also used in a FuseNet.

Comprehensive experimental evaluations demonstrate that the FuseNet performs well on the SUN RGB-D dataset. Segmentation results of FuseNet in comparison to the networks trained with RGB, depth, HHA and their combinations is shown below.

| Input | Global | Mean | IoU |
|---|---|---|---|
| Depth | 69.06 | 42.80 | 28.49 |
| HHA | 69.21 | 43.23 | 28.88 |
| RGB | 72.14 | 47.14 | 32.47 |
| RGB-D | 71.39 | 49.00 | 31.95 |
| RGB-HHA | 73.90 | 45.57 | 33.64 |
| FusetNet-SF1 | 75.48 | 46.15 | 35.99 |
| FusetNet-SF2 | 75.82 | 46.44 | 36.11 |
| FusetNet-SF3 | 76.18 | 47.10 | 36.63 |
| FusetNet-SF4 | 76.56 | 48.46 | 37.76 |
| FusetNet-SF5 | 76.27 | 48.30 | 37.29 |
| FusetNet-DF1 | 73.37 | 50.07 | 34.02 |
| FusetNet-DF2 | 73.31 | 49.39 | 33.97 |
| FusetNet-DF3 | 73.37 | 49.46 | 33.52 |
| FusetNet-DF4 | 72.83 | 49.53 | 33.46 |
| FusetNet-DF5 | 72.56 | 49.86 | 33.04 |

Table 1: Performance of FuseNet on SUN RGB-D dataset with a different parameters

Despite all the above-mentioned results of different models for RGB-D data, recent research [18] shows that the additional information contained in the depth channel does not significantly improve the model performance due to the supreme results of the existing CNN architectures trained just on RGB images. It is getting harder and harder to gain a noticeable extra performance. However, the necessity of using extra channel and the ways of implementing it is still an open question.

During the work the state-of-art Mask R-CNN [4] semantic segmentation CNN was adapted, fine-tuned on the same dataset as RGB-D models for performance comparison. The proposed model was implemented using the TensorFlow framework.

## 3 DATASET

Despite the popularity of CNN models and affordability of the depth sensors, there are still a limited amount of available annotated datasets. Moreover, RGB-D image datasets are still too small for directly training deep CNNs, in contrast to the massive RGB datasets. The most well-known datasets are: NYUv1 [20], NYUv2 [7], SUN RGB-D [5], SUN3D [21], B3DO (Berkeley 3-D Object Dataset) [22] and Stanford 2D-3D dataset [23].

The main focus of the research lies in the field of robotics; thus, the indoor scenes are more preferable for processing. By using indoor scenes there is a higher likelihood of the edges of the scene staying within the effective range of the depth sensor, as opposed to outdoor scenes. In addition to this, the SUN RGB-D dataset was used in FuseNet networks, which in the primary network for comparison. Taking into considerations all the above, NYUv2 [7] and full SUN RGB-D [5] datasets were used for experiments. COCO dataset [6] initiation weights for Mask RCNN model was chosen for transfer learning because this dataset contains almost all needed classes for indoor image segmentation task.

The small dataset NYUv2 contains 1449 annotated RGB-D images. It was split into 80/10/10 train/validation/test sets for the further processing. All images were resized to 1024x1024 shape with adding black padding if needed. The NYU dataset provides labels for 895 objects. This small amount of data and significant number of classes make overfitting unavoidable even for transfer learning approach. To reduce the number of classes, label names and number mapping were used from the previous researches in this area [24]. At the same time 37 and 40 class mapping is also provided in a dataset metadata [7] but was not used for experiments. The target 13 classes for comparison are: bed, books, ceiling, chair, floor, furniture, objects, picture, sofa, table, tv, wall, window. COCO contains 80 classes, including target: bed, book, ceiling-other, ceiling-tile, chair, floor-marble, floor-other, floor-stone, floor-tile, floor-wood, furniture-other, sofa, couch, dining table, table, tv, wall-brick, wall-concrete, wall-other, wall-panel, wall-stone, wall-tile, wall-wood, window, window-blind, window-other. Assuming that COCO contains the same or similar classes, it was expected Mask RCNN to work relatively well. The SUN RGB-D [5] dataset was used for experiments as well. It contains 10335 RGBD images, 5050 of them usually used for test stage and the rest 5285 are usually used for training.

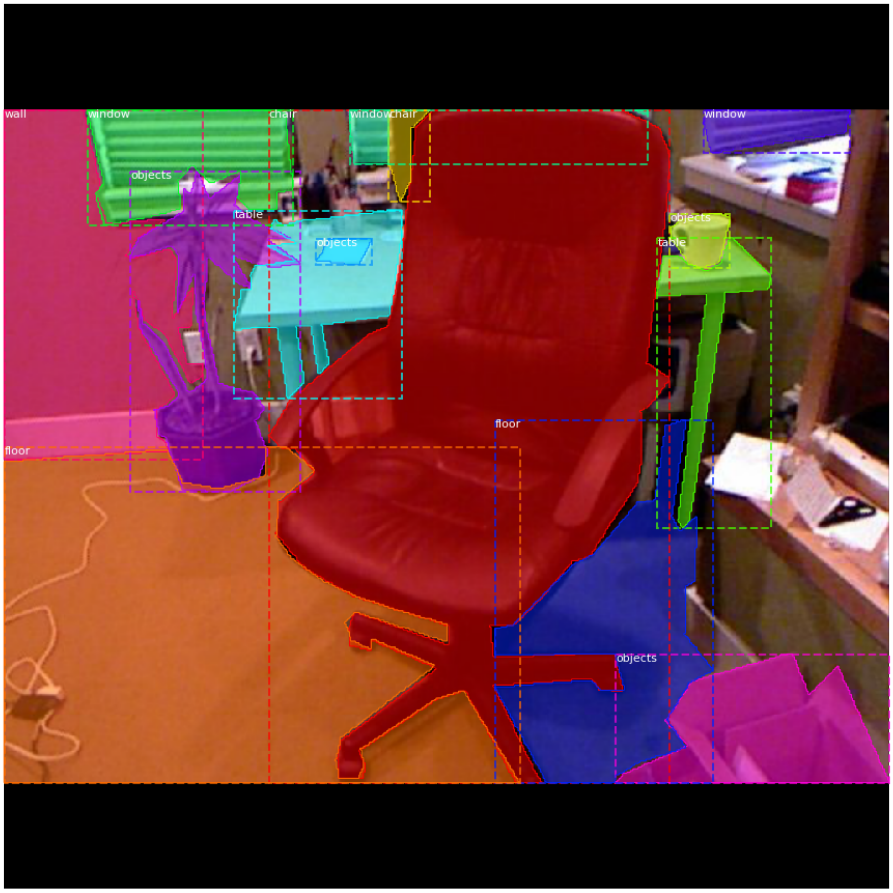


Figure 4: Example of reshaped annotated image with padding.

The main key for working with this dataset is the mapping scheme. Initially the raw data annotations contain 6 585 unique hand-crafted classes, which are impossible to use without any mapping. The creators of this dataset also provided SUN RGB-D Toolbox, which contains the annotation mapping scheme to 40 classes, which requires a RAM of about 64GB to load either in MATLAB or Octave. The obvious way is to avoid the using of semantic segmentation labels stored in this file. Unfortunately, no one provides any kind of mapping from 6 585 classes to 37-40 or 13.

The mapping is stored in an unusual way: there are 2 arrays of arrays both the size of 10 335 (which corresponds to the number of images). In the first array there are 10 335 two dimensional arrays of the size of image, each of them stores in every cell-pixel the corresponding value from 1 to 6 585, which defines the class cell-pixel belongs to. The same with the second array, but the values belongs to 1 to 40 range. After all the parsing it was clear

that only 37 classes out of 6 585 were used for the images description and have a mapping. The parsing, which initially extracted just labels mapping and skipped the pixel-wise mapping was useless, because it requires the images processing, not just working with annotation.

As an alternative solution for mapping 6 585 classes, regular expressions were used for the rude extraction of the class names from the whole annotated name, because some of the hand-crafted annotated classes might be mapped this way without losing semantical meaning. After this, the number of classes was naturally reduced using Levenshtein distance between words or collocation for the class. After that the affinity propagation clustering algorithm was used to see the natural clustering of the set of words, but it identified 100+ classes, which is still a big number of classes, considering the images amount. As a result, the simple k-means algorithm with the 40 classes was applied, as the number of classes was known. Later, the number of classes was reduced to 13, according to the scheme, provided by creators of SceneNetv1.0 model [25]:

| SUN RGB-D/NYUv2 Label Number | Label name | Eigen et al./SceneNet Mapping |
|---|---|---|
| 1 | Wall | 12 |
| 2 | Floor | 5 |
| 3 | Cabinet | 6 |
| 4 | Bed | 1 |
| 5 | Chair | 4 |
| 6 | Sofa | 9 |
| 7 | Table | 10 |
| 8 | Door | 12 |
| 9 | Window | 13 |
| 10 | Bookshelf | 6 |
| 11 | Picture | 8 |
| 12 | Counter | 6 |
| 13 | Blinds | 13 |
| 14 | Desks | 10 |
| 15 | Shelves | 6 |

| | | |
|---|---|---|
| 16 | Curtain | 13 |
| 17 | Dresser | 6 |
| 18 | Pillow | 7 |
| 19 | Mirror | 7 |
| 20 | Floor-mat | 5 |
| 21 | Clothes | 7 |
| 22 | Ceiling | 3 |
| 23 | Books | 2 |
| 24 | Refrigerator | 6 |
| 25 | Television | 11 |
| 26 | Paper | 7 |
| 27 | Towel | 7 |
| 28 | Shower-curtain | 7 |
| 29 | Box | 7 |
| 30 | Whiteboard | 7 |
| 31 | Person | 7 |
| 32 | NightStand | 6 |
| 33 | Toilet | 7 |
| 34 | Sink | 7 |
| 35 | Lamp | 7 |
| 36 | Bathtub | 7 |
| 37 | Bag | 7 |
| 38 | Other-structure | 7 |
| 39 | Other-furniture | 6 |
| 40 | Other-prop | 7 |

Table 2: Mapping scheme for SUN RGB-D dataset

The module to convert SUN RGB-D annotation to VGG-style annotation was created, as well as the tool to map all raw classes to the widely used 13 and 40 classes. This will possibly boost the use of this dataset by other researches.

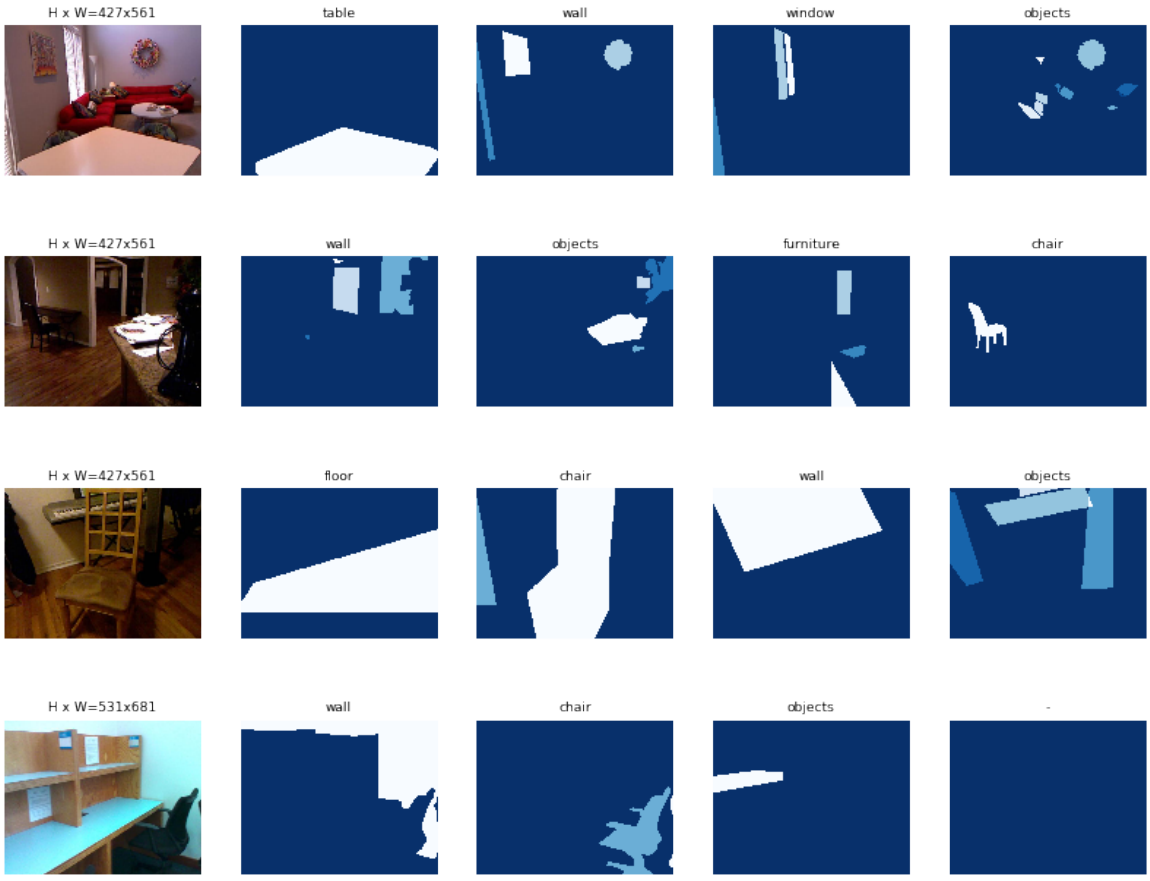


Figure 5: Masks of annotated SUN RGB images

## 4 MEMORY ALLOCATION

A common strategy for improving CNN accuracy is to stack more layers. This might be done via adding more layers, increasing depth or kernel size. The use of this approaches can be seen in a family of ResNets [15]. However, the modern GPUs are still relatively memory constraint. In this project 2 Nvidia 1080Ti GPUs were used with 11 GB of memory.

After starting experimenting, CUDA out of memory error started to appear. Some attempts to deal with memory constraints were taken separately. Modifying the model architecture by changing the number of convolutional layers, trying different optimiser, varying kernel size, changing the activation function, changing number of fully connected layer or increasing the dropout rate was the last wanted thing to do. The initial intention was the testing of the original Mask RCNN network and then fitting it to the dataset. The distribution of the model among multiple GPUs did not help.

The concept of mini-batch in the Fast/Faster/Mask RCNN models differs a bit from the classical understanding of a batch size in a CNNs since different mini-batch sizes are used for different parts of the model. Namely, the network backbone uses one mini-batch size parameter while the network heads use another batch size parameter for the RoIs processing. I changed the batch size for a backbone from initial 4 to 1, aiming it reduce the memory requirements, but it did not help.

Authors of Mask RCNN used 1 GPU with 12GB of memory, so it was estimated that 4 times 11GB of the Nvidia 1080Ti GPUs should be enough for 2-3 people to run training concurrently on the same GPU small or moderate size models. Luckly, it was noticed, that original Mask RCNN model required TensorFlow for CPU, which means that it will most likely fit into the memory of 64 RAM computer. The problem was generated by the TensorFlow, because by default, it allocates the full amount of available memory on the GPU when working. Even for a small Neural Networks all of the 11GB of the Nvidia 1080Ti GPUs were blocked. People usually interrupt processes rather by keyboard interruption, or by killing the processes they see in the 'top' command if the model is stuck on something. Despite everything seems to be clear after running the nvidia-smi command, child processes generated in case of using multithreading or multiple GPUs are still alive and were accidentally tracked using ps command.

Students experiences the situation, when the server was spammed with their's own killed processes, running almost invisible in the background. The root of this problem was mostly because of the lack of communication, as everyone experienced the same error and thought it is something with their model.

To solve this, I contacted other people from my course via a group chat, luckily, we were from the same course, and we killed all extra processes. The second issue is the Python memory management approach itself. Deleting an object in Python does not guarantee releasing

memories, unless the garbage collector is used. People usually not so cautious about the memory until they are run out of it.  is up to TensorFlow to decide what to do.

To sum up, in case of the concurrent access to the GPU, the process is significantly influenced by multitasking, but it is still does make sense to have the flexibility of having several users working on the same server.  Using GPUs is it significantly faster, than training network on CPU. Resource management can be done via several approaches. First, setting the fraction of GPU memory to be allocated when constructing a TensorFlow session by passing an optional configuration argument. This defines the amount of GPU memory that will be used by the process on each GPU on the same machine. The main disadvantage is that it cannot be done in per machine basis. Second, specifying environment variable for visible devices for CUDA helps. It is also possible to grow the memory used on the GPU dynamically, which should be written as a parameter in TensorFlow session. There is currently debates about inclusion of dynamic growth by default because there are always some people who forget to do this. The main difficulties are fragmentation issues. As for today, all GPUs will be used, which causes others not able to calculate, unless the parameters are specified in a code implicitly.

## 5 EXPERIMENTS AND RESULTS

The second version of Mask RCNN, which was used for the experiments consists of Faster-RCNN ResNet-101 and Feature Pyramid network (FPN) [26] backbone. Two models are compared in this research: Mask RCNN architecture using NYU dataset and the full SUN RGB-D dataset, both model's heads were fine-tuned, using a pretrained COCO dataset model. The network was trained with a mini-batch size of 1 and 1 image per GPU for 30 epochs for NYU dataset and 80 epochs for SUN RGB-D dataset.

To adjust a trade-off between accuracy and training time, hyper-parameters were changed: learning rate, regularisation strength. Data augmentation was done as well but was not so useful as expected. The learning rate was changed from original 0.01 to 0.001 for a better

convergence, keeping other parameters the same, optimiser is a SGB plus momentum of 0.9. Weight decay of 0.0001 helped the model to overfit less.

Loss for classification and box regression is same as Faster R-CNN. To each map a per-pixel sigmoid is applied, the map loss is defined as average binary cross entropy loss. Despite the class imbalance is present in both datasets, for instance, wall and object classes are far more popular than TV class. But with this network architecture there is no need to assign the weights for each class, because it optimises each class performance separately and uses the average performance for a loss.

### 5.1 NYU dataset

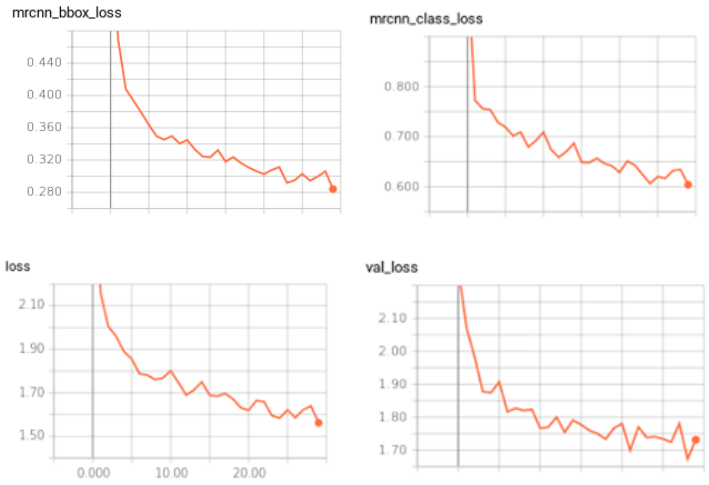


Figure 6: Losses of transfer learning of Mask-RCNN on NYU dataset

The post-processing analysis shows that the training and validation losses significantly go down for this dataset after 7 epochs. 10% mAP value is relatively poor. This might mean, that there is not enough data in the NYU dataset to feed the model, so it simply stops it's improvement.

## 5.2 SUN RGB-D dataset

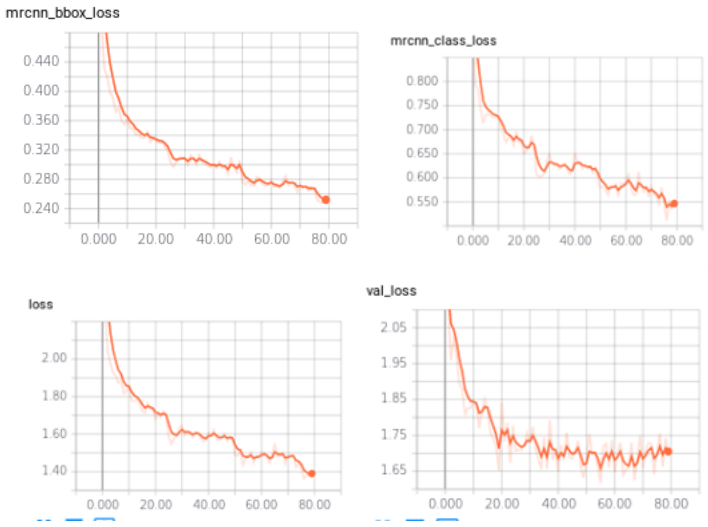


Figure 7: Losses of transfer learning of Mask-RCNN on SUN RGB-D dataset

Comparatively, the SUN RGB-D model performs much better with the same parameters according to the 18% mAP results. This lead me to the conclusion, that size of the dataset matters. Moreover, this dataset was trained for more epochs, meaning that further training of NYU dataset might also lead to the better performance.

Looking at the different model results on a particular objects, it was easy to notice that model catches images, which were in a COCO dataset, but were mentioned in a 7 classes under the general class object. Interestingly, fine-tunes model also finds separate objects, which are even close to each other. In the original annotation provided in SUN RGB-D dataset close objects are not separated as an individual instances. Moreover, model performs well in case when case of overlapping.

A real-time integration of the model to the Kinect v2 video stream is also provided and placed in appendix.

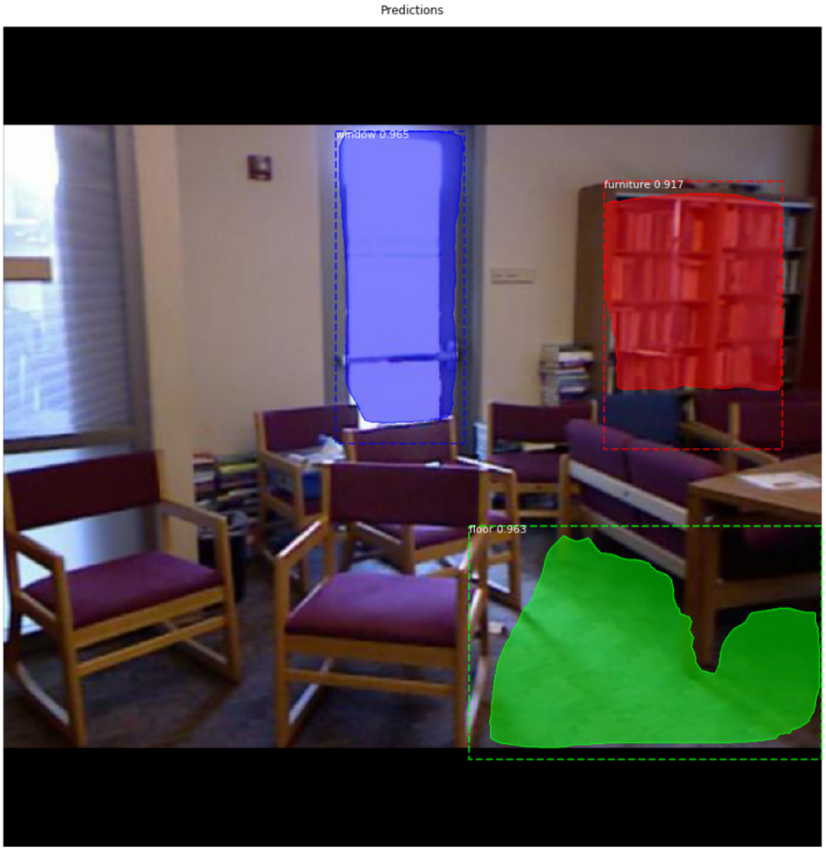


Figure 8: Example of prediction made after transfer learning on an image from the testing part of the dataset

## 6 CONCLUSION AND FUTURE WORK

In this study the ability of Mask RCNN perform on the new dataset was investigated. The performance of the model was compared between NYU and SUN RGB-D dataset. The massive dataset annotation was converted to the most useful VGG annotation format and class mapping to 7 and 37 classes was done, making future work with this dataset more pleasant to the future researchers.

The future work on the model should address the present study. With the small size and objects overlaps NYU dataset seems to be more challenging for tuning. Future hyper parameters tuning and full depth channel incorporation might also benefit the model's performance as well as the general development of deep neural networks. Moreover, 37 class classification will be challenging, because it is more close to the COCOs number of classes.

# 7 References


1. Navneet Dalal, Bill Triggs, Histograms of Oriented Gradients for Human Detection, Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1, p.886-893, June 20-26, 2005
2. D. Lowe, Object recognition from local scale-invariant features, in: ICCV, 1999.
3. H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool. SURF: Speeded up robust features. Computer Vision and Image Understanding (CVIU), 110(3):346-359, 2008. 1, 2, 5
4. He, Kaiming et al. “Mask R-CNN.” 2017 IEEE International Conference on Computer Vision (ICCV) (2017): 2980-2988.
5. B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva Learning Deep Features for Scene Recognition using Places Database Advances in Neural Information Processing Systems 27 (NIPS2014)
6. T.-Y. Lin, M. Maire, S. Belongie et al., “Microsoft COCO: Common Objects in Context.” In European Conference in Computer Vision (ECCV), 2014.
7. Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and.support inference from rgbd images. In ECCV, 2012.
8. Girshick, Ross B. et al. “Rich feature hierarchies for accurate object detection and semantic segmentation Tech report.” (2013).
9. Girshick, Ross B. “Fast R-CNN.” 2015 IEEE International Conference on Computer Vision (ICCV) (2015): 1440-1448.
10. S. Ren, K. He, R. Girshick and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," in IEEE Transactions on Pattern Analysis & Machine Intelligence, vol. 39, no. 6, pp. 1137-1149, 2017.
11. Shelhamer, Evan et al. “Fully Convolutional Networks for Semantic Segmentation.” IEEE Transactions on Pattern Analysis and Machine Intelligence 39 (2015): 640-651.
12. Krizhevsky, I. Sutskever, and G. Hinton. ImageNet classification with deep convolutional neural networks. In NIPS, 2012.
13. Simonyan, Karen & Zisserman, Andrew. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv 1409.1556.
14. C. Szegedy et al., ‘‘Going deeper with convolutions,’’ in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jun. 2015, pp. 1–9.
15. He, Kaiming et al. “Deep Residual Learning for Image Recognition.” 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016): 770-778.

16. Couprie, C., Farabet, C., Najman, L., LeCun, Y.: Indoor semantic segmentation using depth information. CoRR abs/1301.3572 (2013)
17. Hazırbaş, Caner & Ma, Lingni & Domokos, Csaba & Cremers, Daniel. (2016). FuseNet: Incorporating Depth into Semantic Segmentation via Fusion-Based CNN Architecture. 10.1007/978-3-319-54181-5_14.
18. Gupta, S., Girshick, R., Arbelaez, P., Malik, J.: Learning rich features from RGB- D images for object detection and segmentation. In Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T., eds.: Proceedings of European Conference on Computer Vision. Volume 8695 of Lecture Notes in Computer Science., Zurich, Switzerland, Springer (2014) 345–360
19. Holder, Christopher J. et al. "Depth Not Needed - An Evaluation of RGB-D Feature Encodings for Off-Road Scene Understanding by Convolutional Neural Network." CoRR abs/1801.01235 (2017)
20. N. Silberman and R. Fergus. Indoor scene segmentation using a structured light sensor. In International Conference on Computer Vision (ICCV) Workshops, 2011.
21. J. Xiao, A. Owens, and A. Torralba. SUN3D: A database of big spaces reconstructed using SfM and object labels. In International Conference on Computer Vision (ICCV), 2013.
22. A. Janoch, S. Karayev, Y. Jia, J. T. Barron, M. Fritz, K. Saenko, and T. Darrell. A category-level 3-D object dataset: Putting the Kinect to work. In International Conference on Computer Vision (ICCV)Workshop on Consumer Depth Cameras in Computer Vision, 2011
23. Armeni, Iro & Sener, Ozan & R. Zamir, Amir & Jiang, Helen & Brilakis, Ioannis & Fischer, Martin & Savarese, Silvio. (2016). 3D Semantic Parsing of Large-Scale Indoor Spaces. 1534-1543. 10.1109/CVPR.2016.170.
24. D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," in NIPS, 2014
25. Handa, Ankur et al. "Understanding RealWorld Indoor Scenes with Synthetic Data." 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016): 4077-4085.
26. Lin, Tsung-Yi & Dollár, Piotr & Girshick, Ross & He, Kaiming & Hariharan, Bharath & Belongie, Serge. (2016). Feature Pyramid Networks for Object Detection.